预备篇

计算机的基础知识

0.1　40H,62H,50H,64H,7DH ,FFH

0.2　812 ,104, 213, 256, 2936, 941

0.3

十进制数

原码

补码

十进制数

原码

补码

28

1CH

1CH

250

FAH

FAH

-28

9CH

E4H

-347

815BH

FEA5H

100

64H

64H

928

03A0H

03A0H

-130

8082H

FF7EH

-928

83A0H

FC60H

0.4 机器数真值分别为: 27,233,－128,－8,14717,31467,－27824,－12478

0.5 (1) 33H+5AH＝8DH, OV=1, CY=0。 (2) -29H-5DH＝7AH, OV=0, CY=1。

(3) 65H-3EH＝27H, OV=0, CY=1。　　(4) 4CH-68H＝E4H, OV=0, CY=0。

0.6

十进制数

压缩 BCD 数

非压缩 BCD 数

ASCII 码

38

38H

0308H

3338H

255

255H

020505H

323535H

483

483H

040803H

343833H

764

764H

070604H

373634H

1000

1000H

01000000H

31303030H

1025

1025H

01000205H

31303235H

0.7 ASCII 码表示的十六进制数分别为: 105H, 7CAH, 2000H,8A50H

基础篇

第 1 章、MCS-51 单片机结构

1.1　单片微型计算机(即单片机)是包含 CPU、存储器和 I/O 接口的大规模集成芯片,即它本身包含了除外部设备以外构成微机系统的各个部分,只需接外设即可构成独立的微机应用系统。微机处理器仅为 CPU,CPU 是构不成独立的微机系统的。

1.2　参见教材 1.1.1 节

1.3　参见教材第 6 页表格

1.4　参见教材表 1.4

1.5　参见教材表 1.1 和表 1.2

1.6　当 PSW=10H　表明选中的为第二组通用寄器　R0~R7 的地址为 10H~17H

1.7　程序存储器和数据存储器尽管地址相同,但在数据操作时,所使用的指令不同,选通信号也不同,因此不会发生错误。

1.8　内部数据　　程序　外部数据　程序

1.9　振荡周期=0.1667μs　机器周期=2μs　指令周期=2~8μs

1.10　A=0,PSW=0,SP=07,P0~P3=FFH

第 2 章、51 系列单片机的指令系统

2.1　参见教材 2.1 节

2.2　因为 A 累加器自带零标志,因此若判断某内部 RAM 单元的内容是否为零,必须将其内容送到 A,JZ 指令即可进行判断。

2.3　当 A=0 时,两条指令的地址虽然相同,但操作码不同,MOVC 是寻址程序存储器,MOVX 是寻址外部数据存储器,送入 A 的是两个不同存储空间的内容。

2.4

目的操作数	源操作数
寄存器	直接
SP 间接寻址	直接
直接	直接
直接	立即

寄存器间址　　直接

寄存器　　变址

寄存器间址　　寄存器

2.5　Cy=1, OV=0, A=94H

2.6　　√　　×

√　　×

×　　×

√　　√

×　　√

×　　×

×　　×

×　　√

×　　×

×　　×

×　　×

2.7　A=25H　(50H)=0　(51H)=25H　(52H)=70H

2.8　　SP=(61H)　　(SP)=(24H)

SP=(62H)　　(SP)=(10H)

SP=(61H)　　DPL =(10H)

SP=(60H)　　DPH=(24H)

执行结果将 0 送外部数据存储器的 2410 单元

2.9　程序运行后内部 RAM(20H)=B4H,A=90H

2.10

机器码	源程序
7401	LA: MOV A,#01H
F590	LB: MOV P1,A
23	RL A
B40AFA	CJNE,#10,LB
80F6	SJMP LA

2.11 ANL A,#0FH

SWAP A

ANL P1,#0FH

ORL P1,A

SJMP $

2.12 MOV A,R0

XCH A,R1

MOV R0,A

SJMP $

2.13 (1)利用乘法指令

MOV B,#04H

MUL AB

SJMP $

(2) 利用位移指令

RL A

```
RL   A
MOV  20H,A
ANL  A,#03H
MOV  B,A
MOV  A,20H
ANL  A,#0FCH
SJMP $
```

(3)用加法指令完成

```
ADD  A,ACC
MOV  R0,A   ;R0=2A
MOV  A,#0
ADDC A,#0
MOV  B,A    ;B 存 2A 的进位
MOV  A,R0
ADD  A,ACC
MOV  R1,A   ;R1=4A
MOV  A,B
```

```
        ADDC   A,B      ;进位×2
        MOV    B,A      ;存积高位
        MOV    A,R1     ;存积低位
        SJMP   $
```

2.14

```
        XRL    40H,#3CH
        SJMP   $
```

2.15

```
        MOV    A,20H
        ADD    A,21H
        DA     A
        MOV    22H,A    ;存和低字节
        MOV    A,#0
        ADDC   A,#0
        MOV    23H,A    ;存进位
        SJMP   $
```

2.16

```
        MOV A,R0
        JZ     ZE
        MOV    R1,#0FFH
        SJMP   $
ZE:     MOV    R1,#0
        SJMP   $
```

2.17

```
        MOV    A,50H
```

```
        MOV   B,51H
        MUL   AB
        MOV   53H,B
        MOV   52H,A
        SJMP  $
```

2.18

```
        MOV   R7,#0AH
WOP:    XRL   P1,#03H
        DJNZ  R7,WOP
        SJMP  $
```

2.19 单片机的移位指令只对 A,且只有循环移位指令,为了使本单元的最高位移进下一单元的最低位,必须用大循环移位指令移位 4 次。

```
        ORG 0
      CLR C
      MOV A,20H
        RLC A
        MOV 20H,A
        MOV A,21H
        RLC A
        MOV 21H,A
     MOV A,22H
```

```
RLC    A

MOV 22H,A

MOV A,#0

   RLC A

MOV 23H,A

SJMP   $
```

第 3 章、MSC-51 单片机汇编语言程序设计

3.1　　因为是多个单元操作,为方便修改地址使用间址操作。片外地址用 DPTR 指示,只能用 MOVX 指令取数到 A,片内地址用 R0 或 R1 指示,只能用 MOV 指令操作,因此循环操作外部数据存贮器→A→ 内部部数据存贮器。

```
ORG     0000H
                MOV    DPTR,#1000H
                MOV    R0,#20H
LOOP:           MOVX A,@DPTR
                MOV @R0,A
            INC    DPTR
              INC    R0
              CJNE    R0,#71H,LOOP
              SJMP    $
```

3.2　要注意两高字节相加应加低字节相加时产生的进位,同时要考虑最高位的进位。

```
            ORG    0
              MOV    A,R0
              ADD    A,R6
              MOV    50H,A
              MOV    A,R7
              ADDC    A,R1
          MOV    51H,A
          MOV    A,#0
          ADDC    A,ACC
          MOV    52H,A
```

```
SJMP $
```

3.3 A 中放小于 14H(20)的数,平方表的一个数据占 2 个字节,可用 BCD 码或二进制数存放.(如 A 中放的是 BCD 码,则要先化成二进制数再查表。)

```
        ORG     0
            MOV    DPTR,#TAB
            ADD    A,ACC              ;A*2
             PUSH    ACC
            MOVC A,@A+DPTR
            MOV    R7,A
            POP    ACC
        INC    A
          MOVC A,@A+DPTR
    MOV    R6,A
    SJMP    $
TAB: DB 00,00,00,01,00,04, 00,09,00,16H,……
  DB      ……      04H,00
```

3.4 先用异或指令判两数是否同号,在同号中判大小,异号中正数为大.

```
            ORG    0
          MOV    A,20H
          XRL    A,21H
          ANL    A,#80H
          JZ    CMP
```

```
        JB   20H.7,BG
    AG: MOV  22H,20H
        SJMP $
    BG: MOV  22H,21H
        SJMP $
   CMP: MOV  A,20H
        CJNE A,21H,GR
    GR: JNC  AG
        MOV  22H,21H
        SJMP $
```

3.5 fosc=6MHZ

	机器周期数
DELAY: MOV R1,#0F8H	1
LOOP: MOV R3,#0FAH	1
DJNZ R3,$	2
DJNZ R1,LOOP	2
RET	2

(1+(1+2*0xFA+2)*0xF8+2)*12/6MHz

= (1+(1+2*250+2)*248+2)*2us

= 249.494ms

3.6 将待转换的数分离出高半字节并移到低 4 位 加 30H;再将待转换的数分离出低半字节并

30H,安排好源地址和转换后数的地址指针,置好循环次数。

```
        ORG     0000H                          MOV    A,@R0
                MOV    R7,#05H                 ANL    A, #0FH
                MOV    R0,#20H                 ADD    A,#30H
                MOV    R1,#25H                 MOV    @R1,A
NET:            MOV    A,@R0                   INC    R0
                ANL    A,#0F0H                 INC    R1
                SWAP   A                       DJNZ   R7,NE
                ADD    A,#30H                  SJMP   $
                MOV    @R1,A                   END
                INC    R1
```

3.7 片内 RAM 间址寄存器只能有 R0 和 R1 两个,而正数、负数和零共需 3 个寄存器指示地址,这时可用堆栈指针指示第三个地址,POP 和 PUSH 在指令可自动修改地址。R0 指正数存放地址和 R1 指负数存放地址 ,SP 指源数据存放的末地址,POP 指令取源数据,每取一个数地址减 1。

```
        ORG    0000H
        MOV    R7,#10H
        MOV    A,#0                    MOV    @R0,A
        MOV    R4,A                    INC    R0
        MOV    R5,A                    AJMP   DJ
        MOV    R6,A                    NE:    INC    R5
        MOV    R0,#40H                 MOV    @R1,A
        MOV    R1,#50H                 INC    R1
        MOV    SP,#3FH                 AJMP   DJ
```

```
NEXT: POP   ACC                    ZER0:  INC   R6
      JZ    ZER0                   DJ:    DJNZ  R7,NEXT
      JB    ACC.7,NE               SJMP   $
      INC   R4                     END
```

3.8 可直接用 P 标志判断(JB P ,ret)

```
      ORG   0000H
            MOV   A,40H
            JB    P,EN             ;奇数个 1 转移
            ORL A,#80H             ;偶数个 1  最高位加“1”
EN: SJMP  $
```

3.9 取补不同于求补码,求补码应区别正、负数分别处理,而取补不分正、负,因正、负数均有相对于模的补数。用取反加 1 求补也可用模(00H)减该数的方法求补。

```
      ORG   0000H
      MOV   R7,#03H                AB:  INC   R0
      MOV   R0,#DAT A                   MOV   A,@R0
      MOV   A,@R0                       CPL   A
      CPL   A                           ADDC  A,#0
      ADD   A,#01                       DJNZ  R7,AB
      MOV   @R0,A                       SJMP  $
```

3.10 16 个单字节累加应用 ADD 指令而不能用 ADDC 指令,和的低位存 A,当和超过一个字节,和的高字节存于 B,并要加进低位相加时产生的进位,16 个单字节加完后,采用右移 4 次进行除十六求平均值的运算,商在 BUF2 单元,余数在 BUF2-1 单元。

```
ORG 0000H
MOV   R7,#0FH
```

```
MOV    R0,#BUF1
MOV    B,#0
MOV    A,@R0
MOV    R2,A
NEXT:   MOV    A,R2
          INC     R0
          ADD    A,@R0
          MOV    R2,A
          MOV    A,B
          ADDC    A,#0
MOV    B,A
DJNZ R7,NEXT
;以上完成求和

MOV    R6,#04H
MOV    BUF2,A
MOV    BUF2-1,#0
  NEX:       CLR     C
MOV    A,B
RRC    A
MOV    B,A
MOV    A,BUF2
```

```
RRC    A
MOV    BUF2,A
MOV    A,BUF2-1
RRC    A
MOV    BUF2-1,A
DJNZ    R6,NEX
SJMP    $
;以上完成除十六运算
```

3.11 将 20H 单元的内容分解为高 4 位和低 4 位,根据是否大于 9 分别作加 37H 和 30H 处理。

```
ORG    0000H
MOV    A,20H
ANL    A,#0F0H
SWAP    A
ACALL    ASCII
MOV    22H,A
MOV    A,20H
ANL     A,#0FH
ACALL    ASCII
MOV    21H,A
SJMP    $
```

```
ASCII: CJNE    A,#0AH,NE
NE:            JC    A30
ADD    A,#37H
RET
A30:       ADD    A,30H
RET
```

3.12　要注意,位的逻辑运算其中一个操作数必须在 C。

```
ORG    0000H
            MOV    C,20H
            ANL    C,2FH
 CPL    C
            ORL    C,/2FH
            CPL    C
            ANL    C,53H
            MOV    P1.0,C
            SJMP    $
            END
```

3.13

```
                    ORG      0000H
          MOV    C,ACC.3
```

```
ANL   C,P1.4
ANL   C,/ACC.5
MOV   20H,C
MOV   C,B.4
  CPL   C
  ANL   C,/P1.5
  ORL   C,20H
  MOV   P1.2,C
  SJMP  $
  END
```

3.14 设一字节乘数存放在 R1,三字节的被乘数存放在 data 开始的内部 RAM 单元,且低字节存放在低位地址单元,R0 作为被乘数和积的地址指针,用 MUL 指令完成一字节乘一字节,每一次部分积的低位加上一次部分积的高位,其和的进位加在本次部分积的高位上,并暂存,三字节乘一字节共需这样三次乘、加、存操作,以 R7 作循环三次的计数寄存器。

```
        ORG    0000H
      MOV   R7,#03H
      MOV   R0,#data
      MOV   R2,#0
NEXT: MOV   A,@R0
      MOV   B,R1
      MUL   AB
      ADD   A,R2
      MOV   @R0,A
       MOV   A,#0
      ADDC  A,B
            MOV    R2,A
       INC    R0
        DJNZ  R7,NEXT
      MOV   @R0,B
      SJMP  $
      END
```

第 4 章、并行接口 P0-P3 和单片机的中断系统

4.1～4.3　参考教材 4.1 节

4.4　　用 P1.7 监测按键开关,P1.0 引脚输出正脉冲,正脉冲的产生只需要将 P1.0 置零、置 1、延时、再置零即可。P1.0 接一示波器可观察波形。如果再接一发光二极管,可观察到发光二极管的闪烁。电路设计可参考图 4.4

汇编语言程序

```
        ORG   0000H
ABC: CLR   P1.0
     SETB  P1.7
       JB   P1.7,$        ;未按键等待
         JNB   P1.7,$     ;键未弹起等待
     SETB  P1.0
     MOV   R2,#0
DAY:   NOP
       NOP
       DJNZ                                    R2,DAY
```

图 4.4

```
       SJMP   ABC
```

4.5　　电路见图 4.5, 初始值送 0FH 到 P1, 再和 0FFH 异或从 P1 口输出,或使用 SWAP A 指令,然后从 P1 口输出,循环运行,要注意输出后要延时。

汇编语言程序

```
ORG 0000H
     MOV A,#0FH
```

```
ABC:    MOV P1,A
                ACALL D05
                SWAP A
                SJMP ABC
D05:    MOV R6,250
DY:     MOV R7,250

DAY:                                                    NOP
```

图 4.5

```
                NOP
            DJNZ R7,DAY
            DJNZ R6,DY
            RET
            END
```

4.6 如使用共阴极数码管,阴极接地,阳极 a~g 分别接 P0～P3 的某个口的 7 位,将 0～F 的段码列成表,表的内容顺次从该口输出。如数码管接 P3 口。

汇编语言程序

```
        ORG 0000H
        MOV DPTR,#TAB
AGAIN:  MOV R0,#0
NEXT:   MOV A,R0
```

```
        MOVC A,@A+DPTR
                        MOV P3,A
        MOV R7,#0
DAY:    NOP
        NOP
        DJNZ R7,DAY
        INC R0
        CJNE R0,#10H,NEXT
        SJMP   AGAIN
TAB:            DB 3FH,06H…     ;段码表(略)
        END
```

4.7 电路设计见图 4.7,编程如下:

```
        ORG   0000H
       MOV   A,#08H
        MOV   DPTR,#TAB
      MOVC   A,@A+DPTR
            MOV   P1,A
            MOV   R2,#08H
AGAIN:  MOV   A,#01
NEXT:   MOV   P3,A
            ACALL   DAY
```

```
        RL   A
        CJNE  A,#10H,NEXT
        DJNZ  R2,AGAIN
        SJMP  $
TAB:    DB 3FH,06H···
```

图 4.7

```
        END
```

4.8　P1 口的八根线接行线,输出行扫描信号,

P3 口的八根线接列线,输入回馈信号。

见图 4.8。

4.9～4.12 参见 4.2 节

4.13　电路设计见图 4.13

汇编语言程序

```
        ORG 0000H
        AJMP  MAIN
        ORG   0003H
        RL   A          ;中断服务
        MOV   P1,A
        RETI
```

图 4.8

```
MAIN:   MOV   A,#0FEH
        MOV   P1,A    ;第一灯亮
        SETB  EA
```

```
        SETB    EX0
      SETB    IT0
      SJMP    $
```

汇编语言中只有一个中断源,不存在占用别的中断源向量地址问题,程序顺序排下,应注意程序的执行过程。C 语言无循环移位指令移位后,后面补零,因此和 01 相或。

4.14 略

4.15 图 4.13

```
          ORG    0000H
        AJMP    MAIN
        ORG    0003H ;中断服务
        XRL    P1,#0FFH
        DJNZ    R0,NE
    CLR    EA
NE:     RETI
          ORG    0030H
MAIN:   SETB    EA
  SETB    EX0
        SETB    IT0
        MOV P1,#0FFH
  MOV    R0,#0AH
```

```
                SJMP   $   ;等待中断
```

因一亮一灭为一次,所以共十次。

4.16　两个数码管阳极经驱动器接 P1 口,阴极分别接 P3.0、P3.1。

```
                aa EQU 08H          ;存储高四位的段码
                bb EQU 09H          ;存储第四位的段码
                i EQU 0AH           ;存储计数值
Tab:            DB   3FH,06H……     ;段码表略
                ORG   0000H
                AJMP   MAIN
                ORG   0013H
                AJMP   INTR
MAIN:       MOV   DPTR,#Tab
                CLR   A
                MOVC   A,@A+DPTR
                MOV   aa,A
                MOV   bb,A                  ;a=b=Tab[0]
                CLR   P3.0
                CLR   P3.1
                SETB   EA
                SETB   EX0
                SETB   IT0                  ;开中断
LOOP:       SETB   P3.0
                CLR   P3.1
                MOV   P1,bb                 ;显示低位
                ACALL   Delay               ;延时
                CLR   P3.0
```

```
                SETB    P3.1
                MOV     P1,aa                   ;显示高位
                ACALL   Delay                   ;延时
                SJMP    LOOP
INTR:           CLR     EX0
                INC     i                               ;i 加一
                MOV     A,i
                ANL     A,#0FH                  ;取 i 的低位
                MOV     DPTR,#Tab
                MOVC    A,@A+DPTR
                MOV     bb,A            ;查表 b=Tab[i 的低位]
                MOV     A,i
                ANL     A,#0F0H
                SWAP    A                       ;取 i 的高位
                MOVC    A,@A+DPTR
                MOV     aa,A            ;查表 a=Tab[i 的高位]
                SETB    EX0
                RETI
Delay:          ;略
                END
```

提示:将 X1 至 X3 分别接至一个三输入或非门的三个输入端，同时还分别接至单片机的三个 IO 口，或非门的输出端接至单片机的外部中断引脚。中断服务程序中检查三个 IO 口的值，便可知道具体的故障源。程序略。

IO 口，或非门的输出端接至单片机的外部中断引脚。中断服务程序中检查三个 IO 口的值，便可知道具体的故障源。程序略。

第五章、单片机的定时/计数器与串行接口

5.1～5.3　请参考教材

5.4　方式 0:　16.38ms　方式 1:　131ms　方式 2:　512μs

5.5　使用方式 2　计数初值 C=100H-0AH=F6H

查询方式:

```
        ORG   0000H
        MOV   TMOD,#06H
        MOV   TH0,#0F6H
        MOV   TL0,#0F6H
        SETB  TR0
ABC:    JNB   TF0,$
        CLR   TF0
        CPL   P1.0
        SJMP  ABC
```

中断方式:

```
        ORG   0000H
        AJMP  MAIN
        ORG   0000BH
```

```
        CPL   P1.0
        RETI
MAIN: MOV TMOD,#06H
        MOV   TH0, #0F6H
        SETB  EA
        SETB  ET0
        SETB  TR0
              SJMP  $      ; 等待中断
```

5.6 1000HZ 的周期为 1ms,即要求每 500μsP1.0 变反一次,使用方式 T1 方式 1,MC=12 / fosc=1μs, C=216-500μs /1μs =FE0CH,除 TMOD=10H,TH0=FEH,TL0=0CH 外, 程序与 5.5 题相同,注意每次要重置 TH0 和 TL0

5.7 f=6MHz MC=2μs 方式 2 的最大定时为 512μs 合乎题目的要求。50μs 时，计数初值为 C1=256-25=E7H，350μs 时计数初值为 C2=256-175=51H

汇编语言程序

```
              ORG   0000H
        MOV   TMOD,#02H
NEXT:   MOV   TH0,#51H
        MOV   TL0,#51H
        CLR   P1.2
      SETB  TR0
AB1:  JBC   TF0,EXT
```

```
        SJMP   AB1
EXT:    SETB   P1.2
        MOV    TH0,#0E7H
        MOV    TL0,#0E7H
AB2:    JBC    TF0,NEXT
        SJMP   AB2
```

上述的计数初值没有考虑指令的执行时间,因此误差较大,查每条指令的机器周期,扣除这些时间,算得 C=E3H,这样误差较小。

5.8　P1.0 输出 2ms 脉冲,P1.0 输出 50μs 脉冲。

汇编语言程序

```
ORG    0000H
       MOV    TMOD,#02H
       MOV    TH0,#06H
       MOV    TL0,#06H
       SETB   TR0
       MOV    R0,#04H
NE:    JNB    TF0,$
       CLR    TF0
       CPL    P1.1
       DJNZ   R0,NE
       CPL    P1.0
```

```
        AJMP   NE
```

5.9

```
                ORG   0000H
MAIN:       MOV   TMOD,#15H
LOOP:       LCALL   Counter
                LCALL   Timer
                SJMP   LOOP
Counter:    MOV   TH0,#0FDH
                MOV   TL0,#18H
                SETB   TR0
                CLR   TR1
                JNB   TF0,$
                CLR   TF0
                RET
Timer:          MOV   TH1,#0F9H
                MOV   TL1,#30H
                SETB   TR1
                CLR   TR0
                JB   TF1,$
                CLR   TF1
```

RET

END

5.10　略

5.11　参见教材 5.3.1 节 5.125555

5.12　方式 3 为每桢 11 位数据格式

3600*11/60=660(波特)

5.135.13　T1 的方式 2 模式不需要重装时间常数(计数初值),不影响 CPU 执行通信程序.

设波特率为 fbaut 计数初值为 x,

依据公式 fbaut=2somd/32*(fosc/12(256-x))　求得 x=256-((2SMOD/32)*(fosc/fbaut))

5.01…33333333333333355.14　最低波特率为 T1 定时最大值时,此时计数初值为 256,并且 SOMD=0, fbaut=(1/32)*(fosc/(12(256-0))=61

最高波特率为 T1 定时最小值(1)且 SOMD=1 时

fbaut=(2/32)* fosc/(12(256-1))=31250

5…5.15　取 SMOD=1　计算 TH1=TL1=B2

发送

```
ORG   0000H
  MOV   TMOD,#20H
  MOV   TH1,#0B2H
```

```
                    MOV    TL1,#0B2H
                    SETB    TR1
                    MOV    SCON,#40H
                    MOV    A,#0
NEXT:       MOV    SBUF,A
TES:            JBC    T1,ADD1
                SJMP    TES
ADD1:       INC    A
                    CJNE    A,#20H,NEXT
                    SJMP    $
                     END
```

接收

```
                    ORG    0000H
                    MOV    TMOD,#20H
                    MOV    TH1,#0B2H
                    MOV    TL1,#0B2H
                    SETB    TR1
                    MOV    SCON,#50H
                    MOV    R0,#20H
 TEC:        JBC    RI,REC
```

```
        SJMP   TEC
REC:    MOV    @R0,SBUF
        INC    R0
        CJNE   R0,#40H,TEC
        SJMP   $
        END
```

5.16 略

5.17 利用串行通信方式 2(波特率固定),采用奇校验方式,将校验位放在 TB8 中,乙机检验校验位,如正确,则存于片外 4400H 开始的 RAM 中,如错误,通知对方重发,R6 存放数据块长度汇编语言程序如下:

发方

```
        ORG    0000H
               MOV    DPTR,#3400H
               MOV    R6,#0A1H
               MOV    SCON,#90H
               MOV    SBUF,R6
L2:     JBC    T1,L3
        AJMP   1.2
```

```
        L3:     MOV   1,@DPTR
                JB    P,L4
                SETB  TB8
L4:     MOV   SBUF ,A
L5:     JBC   T1,L6
                AJMP  L5
L6:     JBC   RI,L7
                AJMP  L6
L7:     MOV   A,SBUF
                CJNE  A,#0FF0H,L8
                AJMP  L3
        L8:     INC   DPL
                DJNZ  R6,L4
                SJMP  $
```

收方

```
                ORG   0000H
                MOV   DPTR,#4400H
                MOV   SCON,#90H
        L1:     JBC   RI,L2
```

```
                AJMP   L1
L2:     MOV    A,SBUF
                MOV    R6,A
L3:     JBC    RI,L4
                AJMP   L3
L4:     MOV    A,SBUF
```

```
JB    P, L5
                JNB    RB8,L8
                SJMP   $
L5:     JB    JB8,L8
L6:     MOVX   @DPTR,A
          INC    DPL
          INC    DPH
          DJNZ   R6,L3
          SJMP   $
L8:     MOV    A,#0FFH
          MOV    SBUF,A
L9:     JBC    TI,L3
          AJMP   L9
          SJMP   $
```

```
        END
```

5.18 电路图见教材中图 5.18，程序如下：

```
    ORG   0000H
    MOV   R5,#03H
    CLR   A
    MOV   SCON,A
LOOP:    SETB   P3.3
    CLR   A
MOV   R7,A
DEF:    MOV   A,R5                          ;循环 4 次
MOV   DPTR,#tab
MOVC A,@A+DPTR;查表 A=tab[R5]
MOV   SBUF,A
DEC   R5
JNB   T1,$
CLR   T1
CJNE   R5,#0FFH,ABC;若 R5==255，则 R5=7
MOV   R5,#07H
ABC:    INC   R7
CJNE   R7,#04H,DEF    ;循环 4 次
```

```
CLR   P3.3
LCALL   timer
SJMP   LOOP
timer:     MOV A,#64H
FOR:     JZ   ENDD
    MOV   TMOD,#01H
    MOV   TH0,#0D9H
    MOV   TL0,#0F0H
        SETB   TR0
        JNB   TF0,$
        CLR   TF0
        DEC   A
        SJMP   FOR
ENDD:   RET
tab:            DB   0c0H,0f9H,0a4H……;略
        END
```

第 6 章、单片机总线与存储器的扩展

6.1 参见 6.1 节

6.2 6116 为 2KB×8 位 RAM,共 11 根地址线 A0~A10,接线见图 6.2。

图 6.2

6.3　2732 为 4KB×8 位 EPROM,6264 为 8KB×8 位 RAM,因各只有一片,所以各片选 CE 接地,电路见图 6.3。

图 6.3

6.4　6116 为 2KB×8 位 RAM、2716 为 2KB×8 位 EPROM,地址线均为 11 位,地址线接线参见图 6.3。

6.5 电路见图 6.5。

图 6.5

4 片 2764 的 CE 分别接 138 译码器为 y0、y1、y2、y3 端,各片地址为:

2764(4)　0000H~1FFFH

2764(3)　2000H~3FFFH

2764(2)　4000H~5FFFH

2764(1)　6000H~7FFFH

6.6 设计电路见图 6.6。

图 6.6

第 7 章、单片机系统功能扩展

7.1　将图 6.6 中的 2764 去掉，并改“P2.5”为“P2.4”，改“P2.6”为“P2.7”；程序可参考教材中例 7.1。

7.2 请参考上题和题 4.16。将 244 的输入端最低位经过一个上拉电阻接至+5V，经过一个按钮接到地。

7.3 电路图见图 7.3

```
MOV   DPTR,#0CFFBH

MOV   A, #0A2H

MOVX  @DPTR, A
```

7.4 电路与上题类似，程序较简单，略。

7.5　8255A 口、B 口、C 口、控制口地址分别为 7CFFH、7DFFH 、7EFFH、7FFFH,A 口方式 0 输出,C 口输出,控制字 80H。电路见图 7.5

图 7.3

图 7.5

程序：

```
                ORG   0000H
                MOV   DPTR,#7FFH          ;指向控制口
                MOV   A,#80H              ; A 口 B 口均采用基本输出方式
                MOVX  @DPTR,A             ; 写控制字
                MOV   DPTR,#7CFFH
                MOV   A,#0
                MOVX  @DPTR,A             ;清显示
AGAIN:    MOV   R0,#0                     ;R0 存字形表偏移量
                MOV   R1,#01              ;R1 置数码表位选代码
NEXT:       MOV   DPTR,#7EFFH             ;指向 C 口
                MOV   A,R1
                MOVX  @DPTR, A            ;从 C 口输出位选码
```

```
        MOV   A, R0
        MOV   DPTR,#TAB          ; 置字形表头地址
        MOVC  A, @A+DPTR         ; 查字形码表
        MOV   DPTR,#7CFFH        ;指向 B 口
        MOVX  @DPTR, A           ; 从 B 口输出字形码
        ACALL DAY                ;延时
        INC   R0                 ; 指向下一位字形
        MOV   A,R1
        RL    A                  ;指向下一位
        MOV   R1,A
        CJNE  R1,#10H,NEXT       ;六个数码管显示完?
        SJMP  AGAIN
DAY:    MOV   R6,#50             ;延时子程序
DL2:    MOV   R7, #7DH
DL1:    NOP
        NOP
        DJNZ  R7,DL1
        DJNZ  R6,DL2
        RET
TAB1:   DB    6FH,3FH,3FH ,5EH ;“g00d”(9ood)的字形码
```

7.6 提示:EPROM27128O 16KB×8,地址线为 14 根,6264 为 8KB×8 位,地址线为 13 根,电路可参考教材中图 7.3。

7.7 根据电路连线

I/O 口:A 口:FDF8H,B 口:FDF9H,C 口:FDFAH,

命令/状态口:FDFBH.

定时器 TIMEL:FDFCH　　TIMEH:FDFDH

存贮器 RAM :FC00H～FCFFH

第 8 章、单片机典型外围接口技术

8.1 电路参照教材中图 8.7,不同的是将 P2。7 改为 P2。3,先计算各模拟量对应的数字量:

3C 对应的数字量:5V/3V=255/X　　C=153=99H

同样可算得 1V、2V、4V 对应的数字量分别为 33H、66H、CCH

①　　三角波

```
MOV    DPTR,#OF7FFH
NEXT1: MOV    A,#0
NEXT:    MOVX    @DPTR,A
NOP
NOP
INC    A
CJNE    A,#9AH,NEXT
NEXTA:DEC    A
MOVX    @DPTR,A
NOP
```

```
NOP
CJNE    A,#0,NEXTA
SJMP    NEXT1
END
```

②　　方波

4V 对应的数字量为 CCH

```
        MOV    DPTR, #0F7FFH
        MOV    A,#0
NEXT: MOVX    @DPTR,A
ACALL    D2MS
XRL    A,#0CCH
SJMP    NEXT
```

③　　阶梯波

```
MOV    DPTR, #0F7FFH
NEC:     MOV    A,#0
NEXT:    MOVX    @DPTR,A
ACALL D1MS
ADD    A,#33H
CJNE    A,#0FFH, NEXTA
NEXTA:MOVX    @DPTR,A
ACALL    D5MS
SJMP    NEC
```

8.2 电路参考教材中图 8.8,增加一个地址,使用两条输出指令才能输出一个数据,其他同上。

8.3 电路参考教材中图 8.7,地址为 7FFFH。

```
ORG   0000H

MOV   DPTR,#7FFFH

MOV   R0,#20H

MOV   A, @R0

NEXT:   MOV X @DPTR,A

ACALL   D1MS

INC   R0

CJNE   R0,#30H,NEXT

SJMP   $

END
```

8.4 电路参阅教材中图 8.11,不同的是将 P2。5~P2。7 改为 P2。0~P2。2,各地址分别为 FEFFH、FDFFH、FBFFH。程序参照教材 8.1.2.3 节,注意修改 RAM 地址,循环执行该程序。

8.5 电路参阅教材中图 8.2,不同的是 延时方式:EOC 悬空;查询方式:EOC 经非门接单片机 P1.0 端口线;中断方式同原图。

下面仅编查询程序。IN2 的地址为 7FFAH,由于 EOC 经非门接单片机 P1.0 端口线,

查询到 P1.0 为零,即转换结束。

```
ORG 0000H

MOV   R7,#0AH

MOV   R0,#50H

MOV   DPTR,#7FFAH

NEXT: MOVX @DPTR,A   ;启动转换
```

```
JB   P1.0,$              ;查询等待

MOVX   A,@DPTR ;读入数据

MOV   @R0,A

INC   R0

DJNZ   NEXT

SJMP   $
```

8.6　　ADC0809 采集入中模拟信号,顺序采集一次,将采集结果存放于数组 ad 中。ADC0809 模拟通道 0~7 的地址为 7FF8H~7FFFH,ADC0809 的转换结束端 EOC 经逻辑非后接至外部中断 1,电路参考教材中图 8.2。程序参考教材第 167 页的例子，只需修改数据存储区地址。

8.7　　电路参考教材中图 8.26，增加键盘的行线和数码管个数至 8 个，减少键盘列线到 2 根，程序略。

第 9 章、串行接口技术

9.1-9.3 请参考教材

9.4 电路参照教材中图 9.12，另外一片 24C04 的 A1 接到 VCC 其它引脚与第一片完全一样。

9.5 略

9.6 可以，在操作 IIC 总线时，将 SPI 总线上的所有器件的从机选择线置高，这样便不会对 SPI 总线有影响；在操作 SPI 总线时，让 IIC 总线的 SDA 保持高电平，这样 IIC 总线得不到起始信号，便不会对 IIC 总线有影响。

9.7 TLC5615 经 SPI 总线接至单片机（参照教材中图 9.26），REFIN 作为衰减器的输入，OUT 作为衰减器的输出。根据 ,其增益为：　。

9.8 提示：用较快的速度对被测电压进行采样（采样时间间隔恒定为 t），将一定时间段（T）内的获得的采样值（v）的平方对时间积分（实为求和）后除以该时间段的长度，最后开平方，便是被测电压在该时间段内近似的有效值。

，其中 k=T/t。

应用篇

第 10 章、单片机的 C 语言编程——C51

10.1 第 6 行，缺少“;”；第 8 行“;”多余；main 函数最后缺少“}”。

10.2 xdata 的地址范围为 0x0000 到 0xFFFF（共 64K），它需要两个字节记录（ ）。

10.3 char bdata a;

float xdata b;

int xdata * c（注意不要定义为“xdata int * c”或“int * xdata c”,这样 c 为自身在 xdata 区，指向默认区域的 int 型数据的指针，与题意不符）；

10.4 main()

{

int xdata c;

c=DBYTE[0x20]*DBYTE[0x35];

}

10.5 #include <reg51.h>

#include <absacc.h>

sbit P1_0=P1^0;

```
sbit P1_2=P1^2;
sbit P1_3=P1^3;
unsigned char * pData=&DBYTE[0x30];
TLC()
{
    unsigned char k,i;
    unsigned char d=0;
    for(k=10;k>0;k--)
    {
        P1_3=0;
        for(i=8;i>0;i--)

        {
            d<<=1;
            d|=P1_2;
            P1_0=1;
            P1_0=0;
        }
        *pData=d;
        pData++;
    }
}
```

```
main()
{
    P1=0x04;   P1_0=0;   P1_3=1;
    TLC();
    while(1);
          }
```

10.6 略

10.7 略

10.8

```
#include<absacc.h>
main()
{
    char i;
    for(i=0x10;i<0x16;i++)
        DBYTE[i]=XBYTE[i];
}
```

10.9

```
#include<absacc.h>
main()
{
      unsigned int * x,* y,* z;
      x=(unsigned int *)0x20;
      y=(unsigned int *)0x22;
      z=(unsigned int *)0x24;
```

```
        if(*x>*y)
            *z=*x;
        else
            *z=*y;
        while(1);
    }
```

10.10

```
#include <absacc.h>
main()
{
    unsigned char *pBCD=(unsigned char *)0x21;
    unsigned long *pBinary=(unsigned long *)0x30;
    unsigned char *pLen=(unsigned char *)0x20;
    *pBinary=0;
    while(*pLen)
    {
        (*pLen)--;
        *pBinary*=10;
        *pBinary+=*(pBCD+*pLen);
    }   //程序认为 BCD 码是低位放在低地址的
}
```

10.11

```
#include <absacc.h>
main()
```

```
{
    unsigned int *pBinary=(unsigned int *)0x30;
    unsigned char *pBCD=(unsigned char *)0x21;
    unsigned char *pLen=(unsigned char *)0x20;
    *pLen=0;
    while(*pBinary)
    {
        *pBCD=*pBinary%10;
        (*pLen)++;
        pBCD++;
        *pBinary/=10;
    }   //程序将 BCD 码的低位放在低地址
}
```

第 11 章、RTX51 实时操作系统　　略

第 12 章、以 MCU 为核心的嵌入式系统的设计与调试

12.1 参见教材第 12.2 节

12.2 提示,.利用定时/计数器定时 100ms,中断 10 次达 1 秒，满 60 秒，分加 1 秒清 0；满 60 分，小时加 1 分清 0，同时分,秒均有十位数和个位数，按十进制进位，并送显示，显示可采用 6 个数码管(或八个数码管),校对可用按键中断方式或按键的查询进行加 1 校对,用并行口接驱动器(非门或三极管)驱动扬声器进行闹钟,如果采用 89C51/S51 做,由于片内已有程序存

储器,四个口用户均可使用.

12.3 提示,两个定时器同时使用，一个定时器产生节拍,另一个定时器产生音符。

12.4-12.5 略

第 13 章、单片机实验指导　　略